|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное автономное образовательное учреждение высшего образования  «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ» |

Кафедра компьютерных технологий и программной инженерии

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ОТЧЁТ ПО ПРАКТИКЕ  ЗАЩИЩЁН С ОЦЕНКОЙ  Руководитель |  | | | |
| Ст. преп. |  |  |  | М.Д. Поляк |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

ОТЧЁТ ПО ПРАКТИКЕ

|  |  |  |
| --- | --- | --- |
| вид практики | производственная | |
| тип практики | По получению профессиональных умений и опыта профессиональной деятельности | |
| на тему индивидуального задания | | Создание программы для замены |
| строк-шаблонов в документах Word на значения, указанные в | | | |
| конфигурационном файле | | | |

|  |  |
| --- | --- |
| выполнен | Шумковым Демидом Вадимовичем |
| фамилия, имя, отчество обучающегося в творительном падеже | |

|  |  |  |  |
| --- | --- | --- | --- |
| по направлению подготовки | 09.03.04 |  | Программная инженерия |
|  | код |  | наименование направления |
| направленности | 01 |  | Разработка программно-информационных |
|  | код |  | наименование направленности |
| систем | | | |
| наименование направленности | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Обучающийся группы № | 4134к |  |  |  | Д.В. Шумков |
|  | номер |  | подпись, дата |  | инициалы, фамилия |

Санкт–Петербург 2024

**Оглавление**

[Цель работы 3](#_Toc176209624)

[Структура программы 3](#_Toc176209625)

[Импорт библиотек: 3](#_Toc176209626)

[Реализация вспомогательных функций: 3](#_Toc176209627)

[Главная программа: 3](#_Toc176209628)

[Результаты 4](#_Toc176209629)

[Акт 4](#_Toc176209630)

[Заявка 5](#_Toc176209631)

[Вывод программы 5](#_Toc176209632)

[Выводы по работе 6](#_Toc176209633)

[Приложение А 7](#_Toc176209634)

# Цель работы

Целью данной работы является разработка Python-программы для автоматической замены строк-шаблонов в документах формата Word на вычисленные значения, заданные в конфигурационном YAML-файле. Программа должна корректно заменять шаблоны в фигурных скобках {}, используя их как валидные выражения для f-строк Python. Вычисление значений происходит с помощью функции eval. Кроме того, программа должна обрабатывать ошибки при вычислении, а также поддерживать работу с датами через библиотеки babel и datetime.

# Структура программы

## Импорт библиотек:

Импорт всех необходимых библиотек, включая docx, yaml, babel.dates, datetime.

## Реализация вспомогательных функций:

Функция num2str(num): Преобразование числа в строковое представление на русском языке.

Функция load\_yaml(yaml\_file): Загрузка и парсинг YAML-файла, содержащего данные и настройки для работы с документами.

Класс FilesProcessor:

Конструктор класса \_\_init\_\_: Инициализация объектов класса, загрузка данных из YAML-файла, обработка списка файлов для обработки.

Метод process\_word\_file(self, word\_file, output\_file): Загрузка документа Word, замена шаблонов на вычисленные значения, сохранение результирующего файла.

Метод process\_text(self, text): Обработка текстовых строк с использованием eval() и замена шаблонов на вычисленные значения. Обработка ошибок вычисления и замена на пустую строку при необходимости.

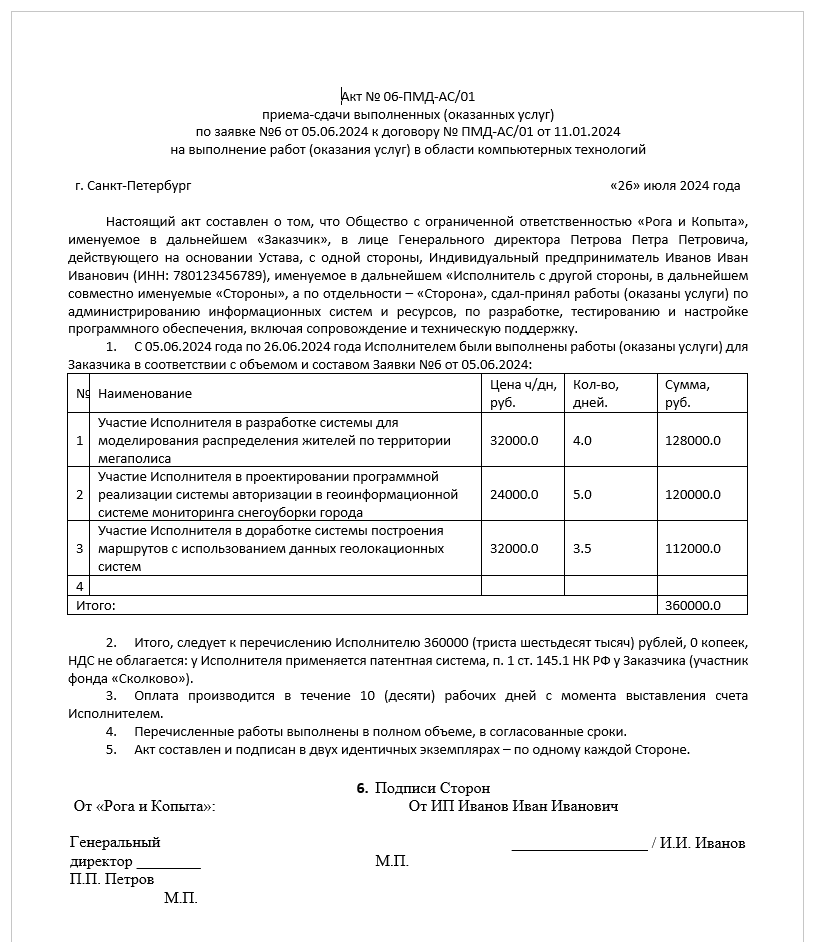
# Главная программа:

Загрузка конфигурационного YAML-файла.

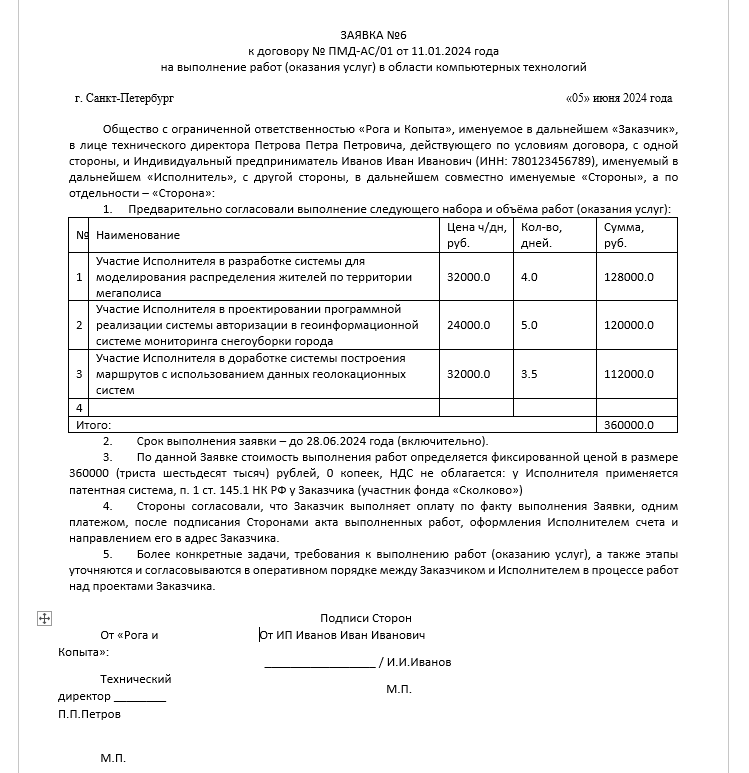
Создание экземпляра класса FilesProcessor и запуск обработки файлов согласно конфигурации.

# Результаты

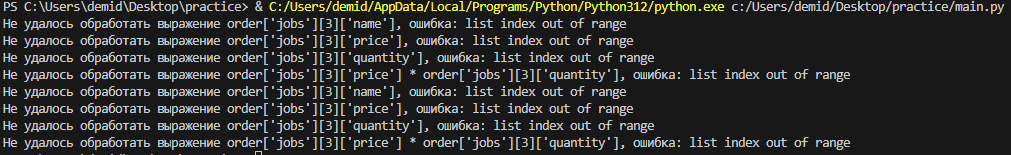
## Акт



## Заявка



## Вывод программы



# Выводы по работе

В ходе выполнения работы была создана программа на Python, способная эффективно обрабатывать шаблоны в документах формата, заменяя их на вычисленные значения из конфигурационного файла.

Обработка шаблонов документов выполняется с использованием библиотеки python-docx для Word.

В коде реализована обработка возможных ошибок при вычислении выражений с информированием пользователя.

# Приложение А

Исходный код программы

import re

import sys

import yaml

from docx import Document

from docx.shared import Cm

from num2words import num2words

import babel.dates

import datetime

def num2str(num):

    if isinstance(num, str):

        num = int(num)

    return num2words(num, lang='ru')

def load\_yaml(yaml\_file):

    try:

        with open(yaml\_file, 'r', encoding='utf-8') as file:

            data = yaml.load(file, Loader=yaml.FullLoader)

    except yaml.YAMLError as exc:

        print(f"Ошибка при чтении YAML файла: {exc}")

        return None

    return data

class FilesProcessor:

    def \_\_init\_\_(self, yaml\_file):

        self.data\_dict = load\_yaml(yaml\_file)

        if self.data\_dict is None:

            sys.exit(-1)

        self.instruments = {

            \*\*{key: self.data\_dict[key] for key in self.data\_dict.keys()},

            \*\*{"num2str": num2str, "babel": babel, "datetime": datetime}

        }

        files\_to\_processing = self.data\_dict["processing"]

        for file\_to\_processing\_key in files\_to\_processing.keys():

            file\_to\_processing\_data = files\_to\_processing[file\_to\_processing\_key]

            file\_template = file\_to\_processing\_data["template"]

            file\_dest = str(self.process\_text(file\_to\_processing\_data['dest']))

            if file\_dest.endswith(".docx"):

                self.process\_word\_file(word\_file=file\_template, output\_file=file\_dest)

            elif file\_dest.endswith(".xlsx"):

                self.process\_excel\_file(excel\_file=file\_template, output\_file=file\_dest)

    def process\_word\_file(self, word\_file, output\_file):

        try:

            document = Document(word\_file)

        except Exception as e:

            print(f"Не удалось открыть файл Word. Ошибка: {str(e)}")

            sys.exit(-1)

        for paragraph in document.paragraphs:

            paragraph.text = self.process\_text(paragraph.text)

        for table in document.tables:

            for row in table.rows:

                for i, cell in enumerate(row.cells):

                    if i == 1:

                        cell.width = Cm(30)

                    for table\_cell\_paragraph in cell.paragraphs:

                        table\_cell\_paragraph.text = self.process\_text(table\_cell\_paragraph.text)

        try:

            document.save(output\_file)

        except Exception as e:

            print(f"Не удалось сохранить заполненный файл Word. Ошибка: {str(e)}")

    def process\_text(self, text):

        def replace\_code(match):

            try:

                return str(eval(match.group(1), self.instruments))

            except Exception as e:

                print(f"Не удалось обработать выражение {match.group(1)}, ошибка: {str(e)}")

                return ""

        return re.sub(r'\{(.\*?)\}', replace\_code, text)

if \_\_name\_\_ == "\_\_main\_\_":

    yaml\_file = 'config.yaml'

    FilesProcessor(yaml\_file)